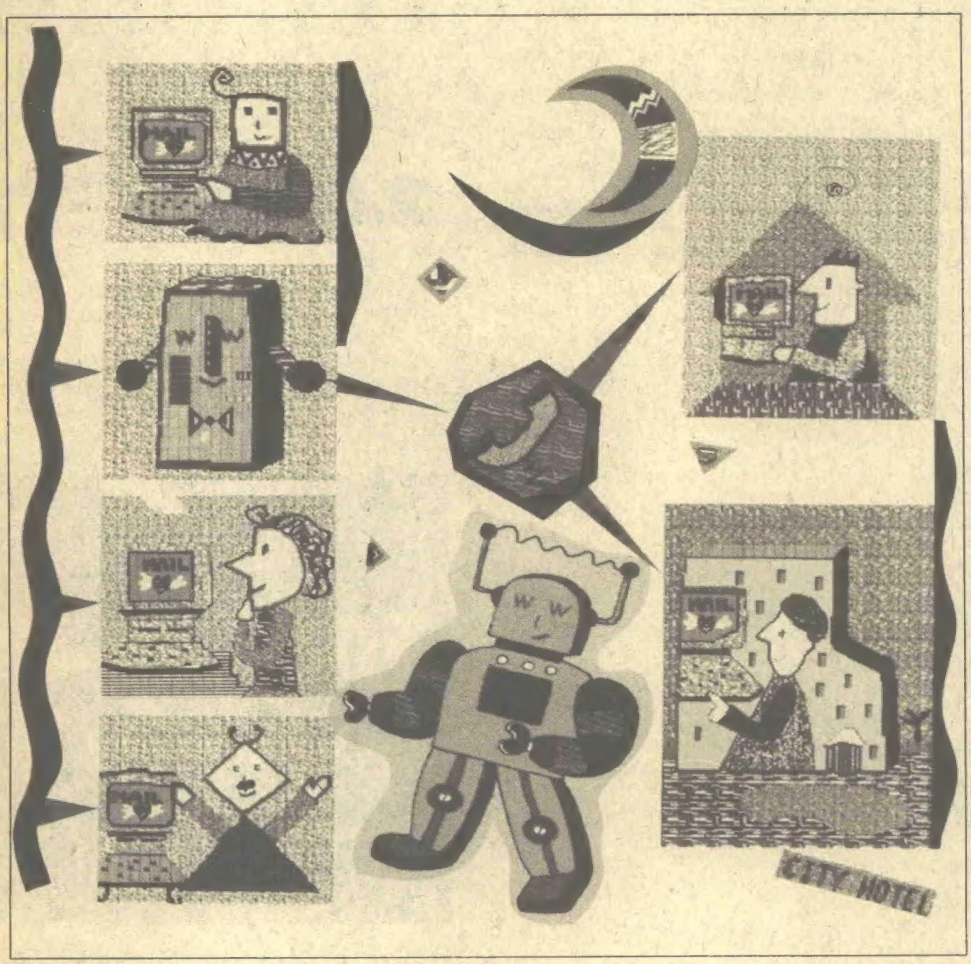
บทความพิเศษ

High Performance Computer

พันธุ์ปิติ เปี่ยมสง่า

แนวคิดหนึ่งสำหรับคอมพิวเตอร์แบบ Connection Machine

เป็นอีกแนวคิดหนึ่งสำหรับระบบคอมพิวเตอร์ประสิทธิภาพสูง Connection Machine



Connection Machine เป็นแนวคิดมาจาก Dr. W.Daniel Hillis ซึ่งการออกแบบ Connection Machine นี้เป็นวิทยานิพนธ์ปริญญาเอกที่ Massachusetts Institute of Technology (MIT) ของเขานั่นเอง การใช้งานของ Connection Machine ในระยะนั้นเน้นหนักในการแก้ปัญหาทางด้านปัญญาประดิษฐ์ เนื่องจากรากฐานความคิดของการออกแบบนั่นคือ ทำอย่างไรที่ทำให้คอมพิวเตอร์คิดเองได้เหมือนมนุษย์

หลังจากที่วิทยานิพนธ์ชิ้นนี้สิ้นสุดในปี 1985 ได้มีการตั้งบริษัท Thinking Machine ขึ้นมาเพื่อขายเครื่อง Connection Machine นี้ โดยมีสถาปัตยกรรมที่มีการปรับปรุงอยู่เรื่อย ๆ โดยจุดที่มีการเปลี่ยนแปลงค่อนข้างมากคือ CM-2 และ CM-5

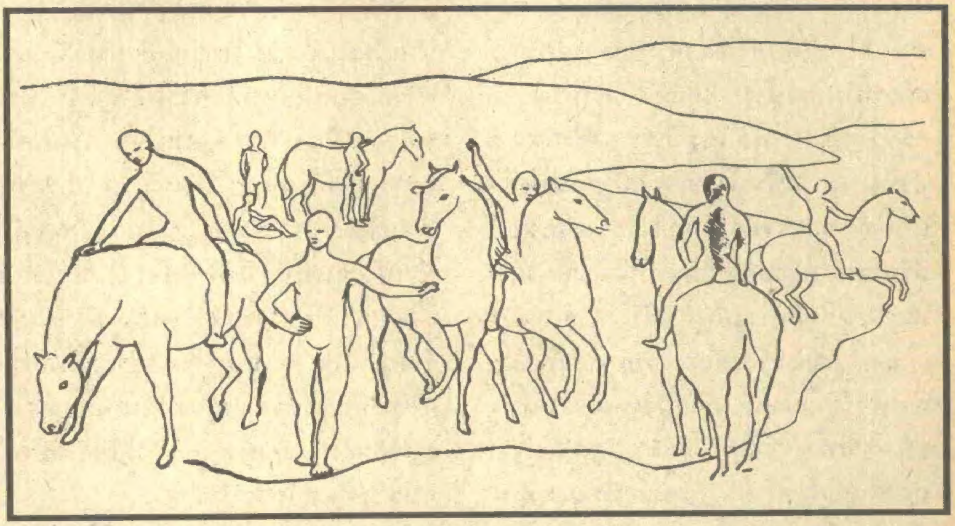
เกริ่นนำ

จากข้างต้นนักคอมพิวเตอร์หลายคนต้องการสร้างคอมพิวเตอร์ที่มีความสามารถที่จะคิดได้เหมือนความคิดของมนุษย์ หรือที่เรียกว่า Thinking Machine ทั้งนี้เนื่องมาจากว่าแนวความคิดที่ออกแบบคอมพิวเตอร์ในปัจจุบันได้ดำเนินไปในทางที่ถูกที่ควรหรือไม่ เครื่องคอมพิวเตอร์ในปัจจุบันมีความสามารถในการประมวลผลข้อมูลจำนวนมากได้ถูกต้องแม่นยำ และ

รวดเร็วกว่ามนุษย์ แต่สำหรับทุกงานหรือเปล่า, คำตอบคือไม่ สำหรับงานที่ซับซ้อนอันเนื่องมาจากอารมณ์ของมนุษย์ แต่บางงานเช่น การตีความหมายรูปภาพระหว่างสมองมนุษย์กับคอมพิวเตอร์อันไหนจะเหนือกว่ากัน

ขอให้ใช้เวลาเพียงแค่นับ 1 ถึง 3 เพื่อดูรูปที่ 1

แม้แต่เด็กเล็ก ๆ จะสังเกตและบรรยายภาพได้เกือบทันทีว่า คือรูป "คนหลายคนอยู่ท่ามกลางม้าหลายตัว โดยคนบางคนขี่ม้า บางคนไม่ได้ขี่ และไกลออกไปมีภูเขาอยู่ 2 ลูก"



รูปที่ 1 ภาพ The Watering Plane, Pablo Picasso, 1905

แต่ถ้าให้คอมพิวเตอร์แปลความหมายนี้จะได้หรือไม่ อย่างไร? คำตอบมีอยู่ว่าอย่างน้อยเครื่องคอมพิวเตอร์จะต้องรับข้อมูลภาพเข้ามา ซึ่งอาจจะมีขนาดของภาพเป็น row และ column ซึ่งทั้ง row และ Column อาจมีค่าสูงเป็นพันค่า นั่นคือ "เม็ดภาพ" ทั้งหมดอาจมีถึง 1 ล้านจุด ในขั้นตอนแรกคอมพิวเตอร์จะหาส่วนที่เป็นเส้นกำหนดบริเวณที่เชื่อมต่อกัน และเงาของภาพ จากรายละเอียดเหล่านี้จะนำมาวิเคราะห์ในโมเดลเชิง 3 มิติ โดยจะได้ผลลัพธ์เป็นวัตถุออกมา และนำวัตถุเหล่านั้นมาเปรียบเทียบกับวัตถุที่มีอยู่ทั้งหมดที่เครื่องรู้จัก ซึ่งย่อมหมายถึงทุกสิ่งทุกอย่างในโลกที่ไม่ใช่เฉพาะเพียงแค่คนกับม้า สุดท้ายก็ย่อมจะได้ผลลัพธ์ออกมาเช่นกัน แต่ก็ไม่ใช่เรื่องง่ายนักกับวัตถุเช่นมนุษย์ที่จะอยู่ในท่าทางเดียวกันทุกครั้ง และย่อมแน่นอน บางครั้งคนขี่ม้าอยู่ และคนยืนอยู่หน้าม้า ก็ย่อมยากต่อการแยกวัตถุของคนกับม้าออกจากกัน

สำหรับงานข้างต้นไม่มีเหตุผลที่แน่นอน เนื่องจากอารมณ์หรือพฤติกรรมอันไม่แน่นอนของมนุษย์มาเกี่ยวข้อง แต่คอมพิวเตอร์ก็ทำงานชนิดนี้สู้มนุษย์ไม่ได้

คอมพิวเตอร์กับสมองมนุษย์

แล้วเหตุไฉนคอมพิวเตอร์จึงสู้สมองมนุษย์ไม่ได้ ปัญหาแรกคือ เราไม่สามารถหาอัลกอริทึมที่เหมาะสมสำหรับงานบางอย่างได้ ซึ่งปัญหานี้พอยอมรับกันได้ แต่ปัญหาที่ 2 คือในเรื่องของความเร็ว ที่บอกว่าคอมพิวเตอร์ช้ากว่านั้นขัดต่อความเป็นจริงอย่างยิ่ง หากเปรียบเทียบระหว่างหน่วยประมวลผลระดับล่างสุดของคอมพิวเตอร์นั่นคือการเปลี่ยนค่าระดับของสัญญาณในทรานซิสเตอร์ที่สามารถทำงานได้ในช่วงเวลาไม่เกินนาโนวินาที แต่สมองมนุษย์มีการเปลี่ยนแปลงในช่วงเวลาที่เป็นหน่วยมิลลิวินาที ย่อมหมายความว่า หน่วยประมวลผลระดับเล็กที่สุดของคอมพิวเตอร์มีความเร็วสูงกว่าสมองมนุษย์อย่างน้อย 1 ล้านเท่า คราวนี้อาจจะมีข้อโต้แย้งว่าสมองมนุษย์อาจเก็บข้อมูลได้มากกว่า แต่ความจริงมีอยู่ว่า ในสมองมนุษย์มี Neural อยู่ในหน่วยของ 10^{10} ตัว แต่ละตัวสามารถเปลี่ยนแปลงได้เกิน 1,000 ครั้งต่อนาที ในขณะที่คอมพิวเตอร์อาจมีทรานซิสเตอร์อยู่ในระดับ 10^9 ตัว แต่ว่าแต่ละตัวสามารถเปลี่ยนแปลงค่าได้ถึง 10^9 ครั้งใน 1 วินาที ดังนั้นหากเปรียบเทียบกำลังในการคำนวณของสมองมนุษย์มีกำลังในการคำนวณเพียง 10^{13} เทียบกับคอมพิวเตอร์อยู่ในระดับ 10^{18} ซึ่งหมายความว่าคอมพิวเตอร์เร็วกว่าถึง 10,000 เท่า ทั้งนี้ยังไม่นับถึงว่าปัจจุบันความหนาแน่นของทรานซิสเตอร์อาจจะมากขึ้นไปอีกก็ได้แต่ Neuron ยากที่จะเพิ่มเติม

แล้วเหตุใดคอมพิวเตอร์ถึงทำงานช้าละ...ยังเป็นคำถามต่อไป

สมมติฐานในความผิดพลาดในการใช้สถาปัตยกรรมคอมพิวเตอร์แบบดั้งเดิม

ผู้ออกแบบ Connection Machine ได้เปรียบเทียบขนาดของเครื่องคอมพิว-

เตอร์ระบบ High Performance ที่มีขนาดใหญ่ด้วยวิธีง่าย ๆ ว่า เครื่องเหล่านั้นต้องการ Silicon ทั้งหมด 1 ตารางเมตร ซึ่งจะมีทรานซิสเตอร์ประมาณ 1 พันล้านตัว โดยส่วนประกอบทุกอย่างไม่ว่าจะเป็นโปรเซสเซอร์หรือหน่วยความจำก็ใช้เทคโนโลยีเดียวกันหมด ซึ่งแตกต่างกับสมัยที่ von Neumann กับลูกทีมออกแบบเครื่องคอมพิวเตอร์เครื่องแรกด้วยวัตถุหลายหลากชนิด ที่ทั้ง "เร็ว" และ "ถูก" เช่น หลอดสูญญากาศ ในขณะที่อุปกรณ์ที่เป็นสารกึ่งตัวนำเป็นชนิดที่ทั้ง "ช้า" และ "แพง" หลอดสูญญากาศในช่วงแรกได้นำมาออกแบบเป็นโปรเซสเซอร์ ในขณะที่หน่วยความจำออกแบบด้วย delay line ทำให้การออกแบบจึงถูกบีบบังคับให้เป็น 2 ส่วน คือส่วนของโปรเซสเซอร์และส่วนของหน่วยความจำ ดังนั้นจึงคาดว่าด้วยเหตุนี้เองที่ทำให้สถาปัตยกรรมแบบ von Neumann จึงต้องแบ่งเป็นส่วน ๆ ตามที่เรารู้จักกัน

จากเทคโนโลยีของวัสดุนี้เองทำให้สถาปัตยกรรมของ von Neumann จึงเป็นอย่างที่เห็น ดังนั้นผู้ออกแบบ Connection Machine จึงพยายามนำประเด็นเหล่านี้มาเกี่ยวข้อง ซึ่งโชคดีที่เทคโนโลยีในวันนี้สามารถทำตามแนวคิดที่อุปกรณ์ทุกอย่างเป็นแบบเดียวกันได้

ปัญหาเนื่องจากประสิทธิภาพการใช้งานของ von Neumann Architecture

จากเครื่องขนาดใหญ่ที่กล่าวมาแล้วว่ามีปริมาณของทรานซิสเตอร์จำนวนมาก ผู้ออกแบบ Connection Machine ได้ตั้งข้อสังเกตว่า การใช้งานค่อนข้างที่จะมีประสิทธิภาพต่ำ เนื่องจากว่าพื้นที่ส่วนใหญ่ของแผ่นซิลิคอนทั้งหมดนั้นคือหน่วยความจำ แต่ในเวลาหนึ่ง ๆ มีการอ้างตำแหน่งจากหน่วยความจำเพียงแค่ 1 ตัวเท่านั้น และงานส่วนใหญ่จะปรากฏบนโปรเซสเซอร์ ซึ่งเป็นพื้นที่ส่วนน้อยประมาณ 2-3 เปอร์เซ็นต์ของพื้นที่ทั้งหมดเท่านั้นเอง โดยทั่วไปอาจกล่าวได้เลยว่า 97 เปอร์เซ็นต์ของทรานซิสเตอร์ในเครื่องที่ใช้สถาปัตยกรรมของ von Neumann อยู่สภาวะที่คอยการทำงาน

ดังนั้นหากออกแบบเครื่องแบบ von Neumann ให้ใหญ่ขึ้น พื้นที่ที่ไม่ใช้งานก็ย่อมมากขึ้น และการทำให้ซีพียูเร็วขึ้นก็ไม่ได้มีประสิทธิภาพเพิ่มขึ้นเท่าใดนัก เพราะสุดท้ายเวลาที่ใช้ส่วนใหญ่ของระบบ von Neumann คือการรับส่งข้อมูลระหว่างโปรเซสเซอร์และหน่วยความจำ

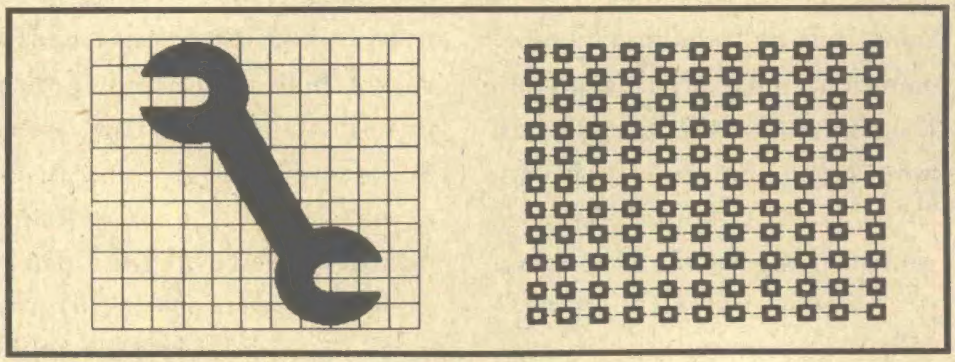
แก้ปัญหาด้วยการทำงานแบบ Concurrence

ในปัจจุบันไม่ใช่เรื่องยากเย็นนักในการสร้างคอมพิวเตอร์ที่ประมวลในเชิงขนานโดยที่มีจำนวนโปรเซสเซอร์เป็นร้อยเป็นพันตัว แต่ปัญหาอยู่ที่เราจะทำงานประยุกต์ประเภทใดใช้กับเครื่องที่เราออกแบบได้บ้าง และปัญหาที่ 2 ที่จะตามมาเมื่อข้อ 1 หมดไปคือจะโปรแกรมงานนั้นอย่างไร เราจะแบ่งงานย่อยของเราออกเป็นร้อยเป็นพันหน่วยย่อยได้อย่างไร และเราจะตามงานย่อยเป็นพันชิ้นรวมกันเป็นงานใหญ่เพียงงานเดียวได้อย่างไร

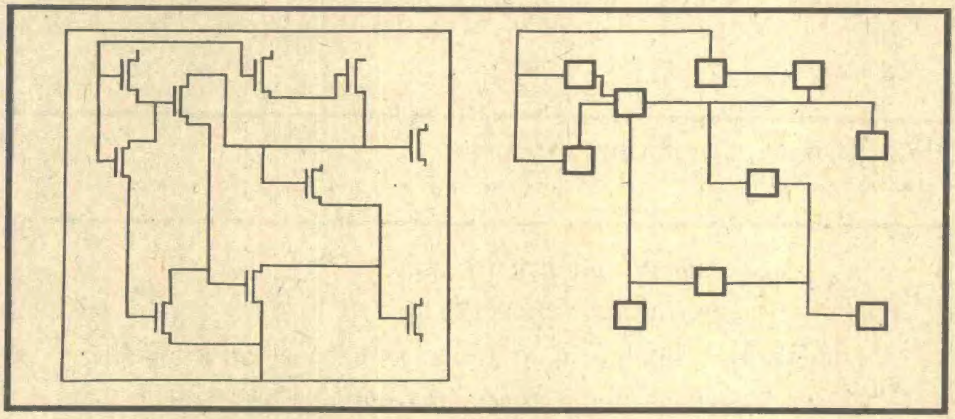
ปัญหาข้างต้นดูจะไม่ใช่เรื่องง่ายเลย แต่เหตุที่ยังคงตั้งไว้เพื่อเป็นสมมติฐานสำหรับวิธีการในการแก้ปัญหาก็เพราะว่า อย่างแรกคือ จะเห็นว่าเครื่องความเร็วสูงแบบเก่าสู้สมองมนุษย์ที่ทำงานช้ากว่าได้ และอย่างที่สองคือ เรามีตัวอย่างของงานที่สามารถทำงานแบบ Concurrence ได้อย่างชัดเจน ดังตัวอย่างที่ยกมา 3 เรื่องต่อไปนี้

1. Image Processing
2. VLSI Simulation
3. Semantic Network

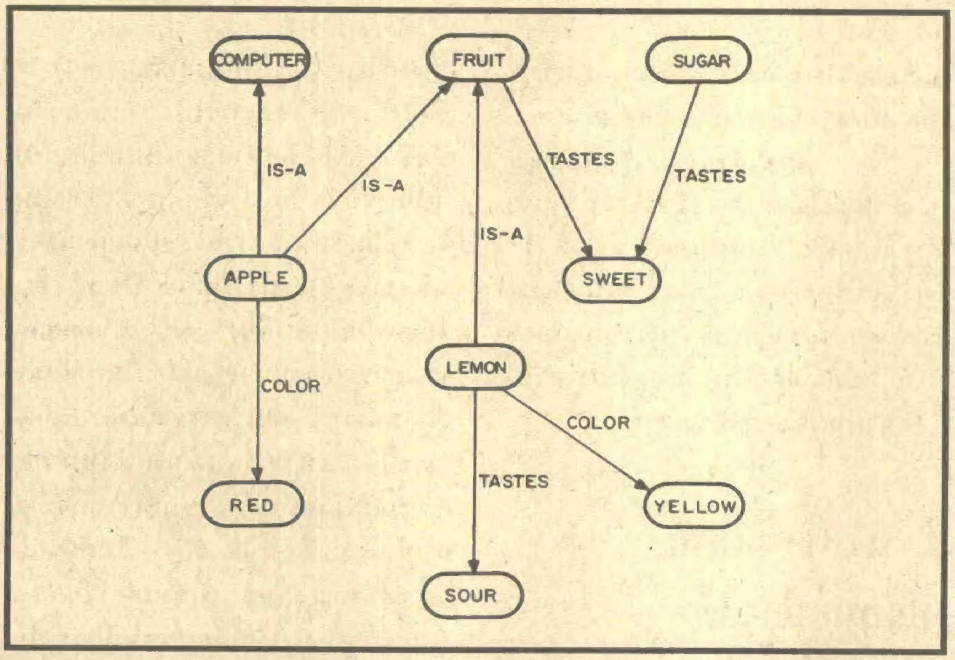
สำหรับงานด้าน Image Processing ดังตัวอย่างในรูปที่ 2 ซึ่งมีรูปภาพอยู่ด้านซ้าย ซึ่งรูปภาพนี้อาจมีขนาด 1000×1000 เราอาจแทนแต่ละพิกเซลของรูปภาพด้วยโปรเซสเซอร์ 1 ตัว ทั้งนี้การประมวลผลที่เหมือน ๆ กัน เช่น การเปรียบเทียบหรือหาค่าเฉลี่ยกับบริเวณข้างเคียงสามารถหาได้ทันที



รูปที่ 2 การแทนพิกเซลด้วยโปรเซสเซอร์โครงสร้างอะเรย์



รูปที่ 3 การแทนทรานซิสเตอร์ด้วยโปรเซสเซอร์ ซึ่งโครงสร้างเป็น Tree



รูปที่ 4 การแทน Semantic Network ด้วยโปรเซสเซอร์ซึ่งมีโครงสร้างเป็นเน็ตเวอร์ก

หากเปรียบเทียบกับการที่มีโปรเซสเซอร์อย่างที่คิด การกระทำด้วย 1 ล้านโปรเซสเซอร์จะทำเร็วกว่า 1 ล้านเท่า (ถึงแม้ว่าการประมวลผลโดยรวมอาจเร็วไม่เท่ากับจำนวนเท่าของโปรเซสเซอร์)

งานด้านที่ 2 คือ การจำลองสถานการณ์ VLSI (VLSI Simulation) ดูในรูปที่ 3 เป็นการจำลองสถานการณ์ของทรานซิสเตอร์แต่ละตัว โดยแต่ละตัวจะทำงานตามสถานการณ์ที่เกิดขึ้นและฟังก์ชันการทำงานที่กำหนด

งานด้าน VLSI Simulation เป็นงานที่อาจไกลตัวสำหรับผู้เริ่มต้นใช้คอมพิวเตอร์ คราวนี้ดูงานประยุกต์แบบที่ 3 คือ การแทนข่ายความหมาย (Semantic Network) โดยแต่ละ Concept แทนด้วยโปรเซสเซอร์หนึ่งตัว ดูรูปที่ 4 เป็นภาพการแสดงความสัมพันธ์ของ Concept ของคำที่มีอยู่ซึ่งมีเป็นหมื่น ๆ คำ และแต่ละคำจะมี link เชื่อม ซึ่งอาจจะมีมากมายมหาศาลอีกเช่นกัน และ link ก็อาจจะไม่ใช้ Is-a ซึ่งเป็น link แบบง่าย ๆ เสมอไป สิ่งที่สลับซับซ้อนย่อมมากขึ้นไปอีก

คำถามง่าย ๆ บางคำถามเช่น แอปเปิ้ลจะตกพื้นหรือไม่ถ้าเราโยนขึ้นฟ้า เป็นคำถามที่แสนจะง่าย กระทั่งเด็กเล็กก็ตอบได้ แต่คอมพิวเตอร์อาจค้นหาในฐานข้อมูลเป็นชั่วโมง

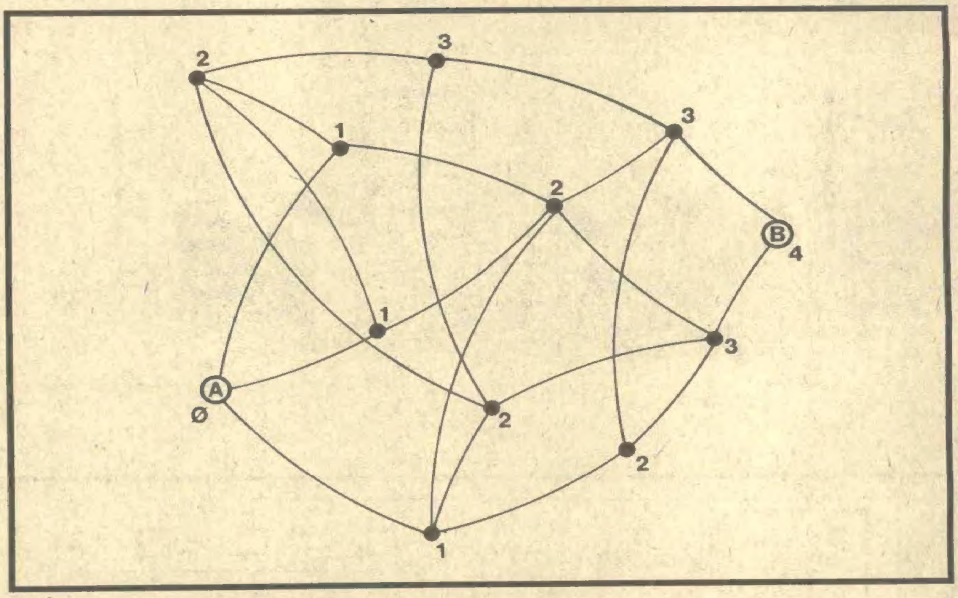
การประยุกต์งานเข้าทำ Connection Machine

มีคำถามอยู่ว่า ในการแก้ปัญหางาน ๆ หนึ่ง เราใช้วิธีการสร้างอัลกอริทึมสำหรับการแก้ปัญหางานชนิดนั้นขึ้นมา มีคำถามถามว่า โดยทั่วไปในการออกแบบอัลกอริทึมนั้นเราพิจารณาถึง

สถาปัตยกรรมคอมพิวเตอร์ด้วยเหรือไม่ ซึ่งคำตอบโดยทั่วไปจะตอบว่า "ไม่" ทีนี้เราก็มักจะพยายามที่จะนำอัลกอริทึมที่เราคิดขึ้นมา มาใช้บนเครื่องที่เป็นสถาปัตยกรรม von Neumann ซึ่งอาจจะไม่เข้ากันเลย แต่เราก็ทนใช้ ยกตัวอย่างเช่น ปัญหา Shortest path ในเน็ตเวอร์ก โดยต้องการหาระยะที่สั้นที่สุดจาก A ไปยัง B ดังรูปที่ 5 ซึ่งเป็นเน็ตเวอร์กตัวอย่าง และรูปที่ 6 คืออัลกอริทึมของการแก้ปัญหา

ในเครื่องความเร็วต่ำอย่าง von Neumann Machine จะแก้ปัญหาได้นานมาก มีการทดลองโดยใช้ VAX-11/750 แก้ปัญหาที่ใช้เวลา 1 ชั่วโมง

การแก้ปัญหานี้อาจเปลี่ยนอัลกอริทึมเพื่อทำงานให้เหมาะสมสำหรับ von Neumann Machine เป็นการเฉพาะก็ย่อมทำได้ แต่ก็ถามว่าจะทำไปทำไมในเมื่อมีเครื่องที่ดีกว่า ทีนี้จากอัลกอริทึมในรูปที่ 6 เรามาออกแบบเครื่องที่เหมาะสมสำหรับอัลกอริทึมนี้ดีกว่า โดยสถาปัตยกรรมควรจะเป็นแบบนี้



รูปที่ 5 โจทย์ตัวอย่างปัญหาการหาระยะระหว่างจุด

1. กำหนดทุกโหนดเป็นระยะอนันต์
2. กำหนดโหนด A มีระยะเป็น 0
3. ทุก ๆ โหนดยกเว้น A, ให้บวก 1 กับค่าต่ำสุดของจุดใกล้เคียงและตัวของมันเอง จนกระทั่งค่าโหนด B เป็นค่าที่ Finite
4. หยุดการทำงาน ค่าที่ B คือคำตอบ

รูปที่ 6 อัลกอริทึมการหาระยะทางระหว่างจุด

1. มากด้วยโปรเซสเซอร์

จากอัลกอริทึมในการหาระยะทางระหว่างจุดจะเห็นว่า งานในหลายส่วนทำงานกันในเชิงขนานได้ แต่ปัญหาอยู่ที่ว่าเครื่องควรจะมีโปรเซสเซอร์กี่ตัวถึงจะพอ เราไม่สามารถบอกได้ว่าเครื่องทำงานแบบขนานควรจะมีโปรเซสเซอร์กี่ตัว เช่นเดียวกับเครื่องแบบคอนเวนชันนัลควรจะมีหน่วยความจำอยู่เท่าไร ทั้งนี้เพราะว่าการเลือกขนาดของเครื่องควรขึ้นอยู่กับปัญหาที่ต้องการ ดังนั้นหากมองในทำนองเดียวกันเครื่องที่เป็นคอนเวนชันนัลจะบอกว่า เครื่องจะต้องมีหน่วยความจำมากเพียงพอที่จะโหลดโปรแกรมขึ้นมาทำงาน ดังนั้น Connection Machine ก็ควรมีโปรเซสเซอร์เพียงพอสำหรับการทำงานเช่นกัน

ทีนี้อาจมีคำถามใหม่ว่า Connection Machine ควรมีหน่วยความจำสำหรับแต่ละโปรเซสเซอร์เท่าไรถึงจะพอ จากการออกแบบเราใช้แนวคิดว่าโปรเซสเซอร์มากหรือหน่วยความจำมาก เมื่อเลือกโปรเซสเซอร์มากทำไมจึงต้องมีหน่วยความจำมากด้วย

2. การเชื่อมต่อที่เปลี่ยนแปลงได้

จากอัลกอริทึมในการหาระยะทางระหว่างจุดต้องแสดงให้เห็นว่า เมื่อเราให้โปรเซสเซอร์แทนแต่ละจุด การเชื่อมระหว่างจุดควรเปลี่ยนแปลงได้เมื่อข้อมูลของโจทย์เปลี่ยนไป ทีนี้คำถามใหม่อาจมีว่า ในการออกแบบเราจำเป็นต้องเชื่อมต่อขาอย่างถาวรบนแผงวงจรแล้วเราจะโปรแกรมได้อย่างไร คำตอบก็ยังเหมือนเดิมคือ von Neumann Machine ของหน่วยความจำในตำแหน่งที่แตกต่างกัน เช่น อาจจะเป็น virtual Memory การเรียกตำแหน่งหน่วยความจำตำแหน่งหนึ่ง เช่น ตำแหน่งที่ 4 กับตำแหน่งที่ 200,000 ไม่มีความแตกต่างกันสำหรับผู้เขียนโปรแกรม นั่นคือ เราสามารถเขียนซอฟต์แวร์เป็นชั้น (layer) ไว้ไม่ให้ผู้เขียนโปรแกรมเห็นการทำงานในระดับล่างเพื่อความสะดวกใน

ค.ศ. 1987

โลกคอมพิวเตอร์รู้จัก FILE SERVER เป็นครั้งแรก

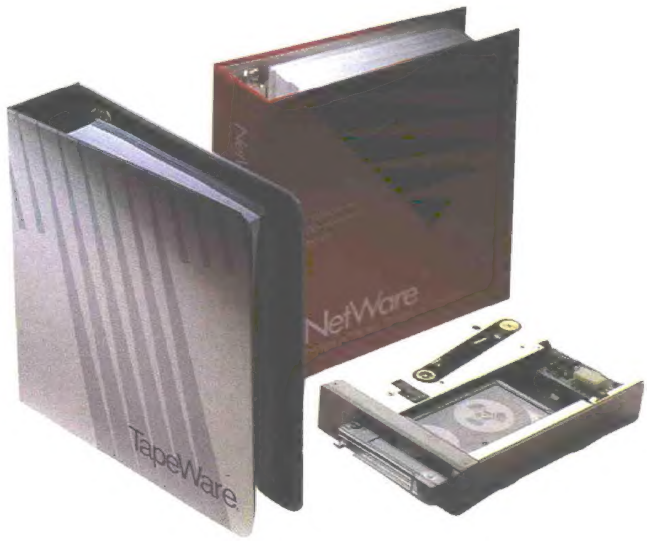
ค.ศ. 1989

คุณรู้ถึงประโยชน์ของ PRINT SERVER แล้ว

ค.ศ. 1993

TAPE SERVER จะช่วยให้การทำงานของระบบ LAN สมบูรณ์แบบที่สุด

TapeWare-TAPE SERVER วิวัฒนาการล่าสุดเพื่อระบบ LAN



TAPE SERVER คือวิวัฒนาการล่าสุดที่พัฒนาขึ้นเพื่อรองรับการทำงานในระบบ LANให้ผู้ใช้ได้รับความสะดวก ปลอดภัยต่อการเก็บรักษาข้อมูล ประหยัดเวลาและคุ้มค่าต่อการลงทุน

เลือก TAPE SERVER ของ TapeWare ที่มีคุณสมบัติพิเศษเฉพาะคือระบบ Library ที่สามารถช่วยค้นหา File ทั้งหมด โดยไม่ต้องอ่านเทปทีละม้วน เป็นที่ยอมรับว่าใช้ได้ดีที่สุดกับระบบ NetWare

PC Computing HARDWARE 200 1993 TOP RATED BEST NETWORK TAPE SOFTWARE

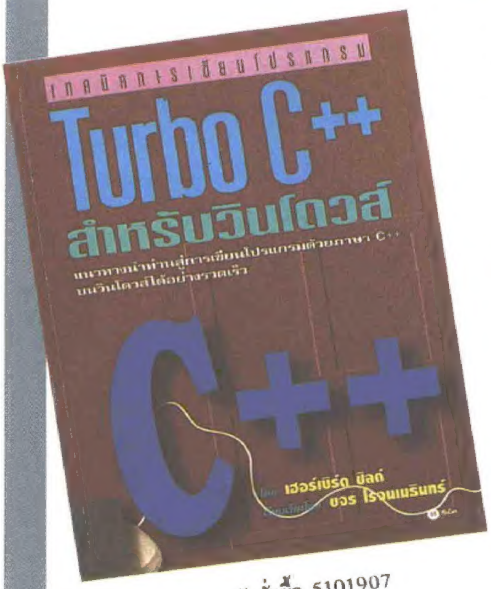
yes It runs with NetWare

TapeWare ผู้นำของระบบ TAPE SERVER

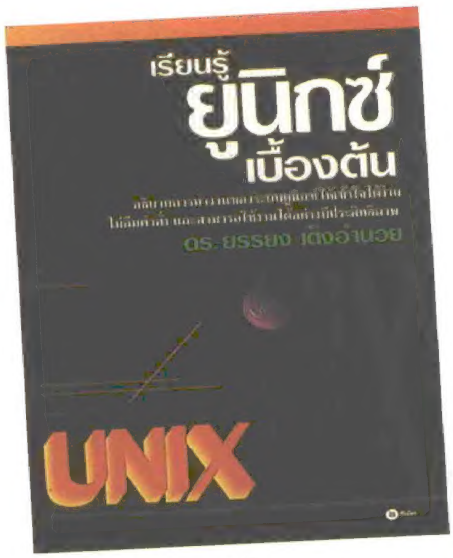
SEACOM

ติดต่อสอบถามเพื่อขอรายละเอียดเพิ่มเติมได้ที่
บริษัท ซีคอม จำกัด (Exclusive Distributor) (อาคาร ซีคอน-นันยาง) 107-115 ถ.สี่พระยา บางรัก กรุงเทพฯ 10500
โทร : 2372900 (อัตโนมัติ 10 เลขหมาย) Fax: 2663960 Phonelink (ฝากข้อความ) : 151,152 เรียก 325531

NEW book



รหัสสั่งซื้อ 5101907
ราคาเล่มละ 240 บาท



รหัสสั่งซื้อ 5101354
ราคาเล่มละ 140 บาท

รหัสสั่งซื้อ 5101176
ราคาเล่มละ 230 บาท

วางตลาดแล้ว

ใบสั่งซื้อ

ข้าพเจ้า .....

ที่อยู่ในการจัดส่ง. .....

. .....

ขอสั่งซื้อ ❑ เทคนิคการเขียนโปรแกรม Turbo C++ สำหรับวินโดว์ ราคา240 บาท จำนวน ..... เล่ม

❑ เรียนรู้ยูนิกซ์เบื้องต้น ราคา 140 บาท จำนวน ..... เล่ม

❑ ไมโครคอนโทรลเลอร์ MCS-51 ราคา 230 บาท จำนวน ..... เล่ม (*สมาชิกวารสารซีเอ็ด ลดพิเศษ 10 % ค่าส่งฟรี หมายเลขสมาชิก* .....)

พร้อมกันนี้ได้ส่งค่าหนังสือจำนวนเงิน ..... บาท มาด้วย

โดยชำระเป็น ❑ เช็ค ❑ ธนาณัติ หรือ ตั๋วแลกเงิน สั่งจ่ายในนาม บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน) ณ ที่ทำการไปรษณีย์โทรเลขดินแดง

บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน) 800/43-45 ซอยตระกูลสุข ถนนอโศก-ดินแดง ห้วยขวาง กรุงเทพฯ 10400 โทร 2486280 แฟกซ์ 2465999

การใช้งานได้นั่นเอง

ความสามารถในการเปลี่ยนวิธีการเชื่อมต่อของโปรเซสเซอร์เพื่อให้เหมือนกับวิธีการเชื่อมโยงในการแก้ปัญหาของโจทย์ต่าง ๆ นี้ เป็นจุดเด่นที่สุดของสถาปัตยกรรมชนิดนี้นั่นเอง นี่คือเหตุของชื่อ Connection Machine

สถาปัตยกรรม Connection Machine

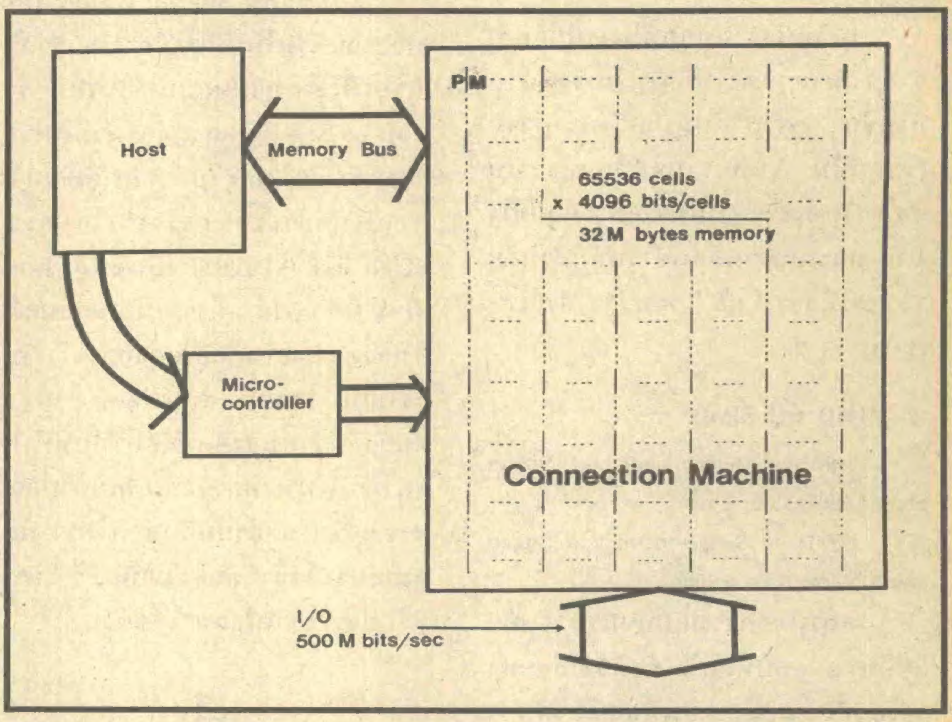
จากอัลกอริทึมหาระยะระหว่างจุดข้างต้น เราได้ข้อกำหนดของ Connection Machine มา 2 ข้อคือ

1. จะต้องมีจำนวน PE (Processing Element) เพียงพอกับขนาดของปัญหา

2. แต่ละ PE เชื่อมโยงกันด้วยซอฟต์แวร์

สรุปออกมาได้ว่า Connection Machine คือเครื่องจักรที่มีโปรเซสเซอร์จำนวนมากเชื่อมโยงกันด้วยซอฟต์แวร์ และโดยภาพแล้วรวมจำนวนโปรเซสเซอร์ต่อขนาดของหน่วยความจำมีค่าต่ำ ในแต่ละปัญหาสามารถแทนการประมวลผลทุกขั้นตอนในเชิงที่ขึ้นกับชนิดข้อมูล ซึ่งโครงสร้างนี้เรียกว่า Active Data Structure

กิจกรรมใด ๆ ภายใน Connection Machine จะได้รับคำสั่งมาจากเครื่องที่เป็น host ซึ่งโดยทั่วไปเป็นเครื่องแบบ Conventional วิธีการนำปัญหาลง Connection Machine ใช้วิธีการเดียวกันกับเครื่องแบบ Conventional คือ นำข้อมูลลงตำแหน่งหน่วยความจำที่กำหนด สำหรับ Connection Machine ทำการประมวลผลข้อมูลพร้อม ๆ กันทุกโปรเซสเซอร์ แต่ถ้าเป็น Conventional ก็ค่อย ๆ ทำทีละงาน



รูปที่ 7 แผนภาพของเครื่องต้นแบบ Connection Machine CM-1

ในรูปที่ 7 เป็นไดอะแกรมของ CM-1 ซึ่งเป็นเครื่องต้นแบบของ Connection Machine

ในเครื่องต้นแบบ CM-1 เป็นเครื่องที่มี 65,536 PE (Processor/Memory Cell) แต่ละ PE มีหน่วยความจำ 4 กิโลไบต์ จากรูปที่ 7 การควบคุมแต่ละ PE จะมาจาก Host Computer

ตั้งแต่เครื่อง CM-1 ได้มีการพัฒนามาเป็น CM-2 ในปัจจุบันถึง CM-5 CM-2 และ CM-5 เป็นเครื่องที่ขายโดยบริษัท Thinking Machine ไม่ใช่สถาบัน MIT

แนวทางที่สำคัญในมุมอื่นสำหรับการออกแบบ

1. Communication

การเชื่อมต่อระหว่างโปรเซสเซอร์มีผลต่อความเร็วและความอ่อนตัวของเครื่องค่อนข้างมาก การกำหนดตายตัวระหว่างจุดเชื่อมซึ่งมักทำกันมากในเครื่องแบบ Massively Parallel จะทำให้ยากต่อการเขียนโปรแกรม

2. จำนวนโปรเซสเซอร์

การแบ่งงานออกเป็นงานย่อย ๆ จำนวนมากจะทำให้มีงานในเชิงขนานมากขึ้น แต่ไม่ได้หมายความว่าความเร็วสูงขึ้น เนื่องจากจะต้องมีระบบซิงโครไนซ์ระหว่างโปรเซสเซอร์ค่อนข้างมาก และเป็นงานค่อนข้างหนักสำหรับระบบที่มีการ "คุย" ระหว่างโปรเซสเซอร์มาก ๆ

3. MIMD กับ SIMD

MIMD = Multiple Instruction Strean and Multiple Data Strean

SIMD = Single Instruction Strean and Multiple Data Strean

สถาปัตยกรรมแบบขนานโดยทั่วไปคือ 2 แบบหลักนี้ ข้อดีข้อเสียของเครื่องทั้ง 2 แบบเป็นที่ถกเถียงกันว่าแบบใดจะดีกว่ากัน สำหรับเครื่องที่เป็น MIMD มีข้อดีคือ แต่ละโปรเซสเซอร์สามารถรันโปรแกรมที่ต่างกันได้ ในขณะที่ SIMD จะต้องรันโปรแกรมตัวเดียวกันสำหรับทุกโปรเซสเซอร์ แต่ว่า MIMD จะต้องมีการซิงโครไนซ์ระหว่างโปรเซสเซอร์ ซึ่งจะเป็นตำแหน่งทำให้ทำงานช้าลงและการทำโปรแกรมตรงส่วนนี้ แต่ SIMD ไม่ต้องมี เพราะมีส่วนที่เป็น Control( )Unit คอยควบคุมอยู่แล้ว

ทางเลือกที่ 3 คือ สามารถโปรแกรมระบบให้เป็น SIMD หรือ MIMD ก็ได้ ซึ่งก็หมายถึงตัว Connection Machine นี้เอง

สรุป

สำหรับ Connection Machine ก็เป็นเครื่องที่มีประสิทธิภาพสูง พร้อมทั้งราคาสูง การใช้งานเครื่องชนิดนี้ยังมีปัญหาพอ ๆ กับการออกแบบของมัน

ในปัจจุบัน ขณะนี้ Connection Machine รุ่นใหม่ที่สุดคือ CM-5 เป็นเครื่องที่ใช้ซีพียูเป็นแบบ SPARC จำนวน 65,536 ตัว และแต่ละตัวมีหน่วยความจำ 32 เมกะไบต์ ราคาผู้เขียนไม่ทราบ แต่ประมาณการจากราคาเครื่อง SUN ที่ใช้ SPARC เป็นซีพียู โดยมีเพียง 1 ตัวเท่านั้น ก็ราคาหลายแสนแล้ว ถึงเครื่องจะแสนดีแสนเก่งอย่างไร ผมยังไม่เห็นว่าเหมาะสมกับราคา เพราะผมยังคิดว่ามันยังคิดอัลกอริทึมที่ทำให้คนไทยทั้งประเทศรวยขึ้นสักนิดไม่ออกหรอกครับ แต่ถ้ามันคิดออก ผมว่ามันคงตอบว่าก็อย่าซื้อของไม่เป็นประโยชน์หรือประโยชน์น้อยมาใช้ซิครับ

ขจัดความกลัว
ในการใช้
เครื่องคอมพิวเตอร์
เครื่องแรก
ของท่านได้แล้ว

ราคาเล่มละ 295 บาท

ไม่ว่าท่านกำลังจะคิดซื้อคอมพิวเตอร์เครื่องแรก หรือเผชิญกับเครื่องคอมพิวเตอร์เครื่องใหม่ในสำนักงานของท่าน หรือแม้แต่ต้องการที่จะรู้ว่า เพื่อนร่วมงานของท่านที่กำลังใช้เครื่องคอมพิวเตอร์อยู่พูดถึงเรื่องอะไร หนังสือเล่มนี้จะช่วยท่านได้

ท่านจะพบกับ...

* คำจำกัดความอย่างชัดเจนถึงศัพท์ที่ใช้อยู่ในวงการคอมพิวเตอร์

* รายละเอียดเกี่ยวกับซอฟแวร์อย่างเป็นขั้นตอน พร้อมด้วยคำแนะนำของผู้เชี่ยวชาญในการเลือกเครื่องมือ

หาซื้อได้ตามร้านหนังสือชั้นนำทั่วไป

มีจำหน่ายแล้ว!